- The second idea behind the FFT concerns how to get the correct h_k's into the final DFT's of size 2. If the subscripts 0 to 7 of the h_k's are written in binary form, as shown in first row of the following table, and the bits are *reversed*, then we get the correct ordering of the subscripts of the h_k's.

000=0	001=1	010=2	011=3	100=4	101=5	110=6	111=7
000=0	100=4	010=2	110=6	001=1	101=5	011=3	111=7

On the next page is a subroutine, *FFT*, written in Visual Basic and based on one given in *Numerical Recipes* by Press, Flannery, Teukolsky and Vetterling that implements the FFT. The input quantities to *FFT* are:

- *ISign*, normally set to 1, is the sign in the exponential of Eq. (10.88). If *ISign* is set to –1, the routine calculates the inverse transform (10.90) – except that it does not multiply by the factor l/*N*. You have to do that yourself.
- The number of data points, *N*.
- The array, *Data*, of length 2*N*. Upon entry it contains the h_k's. The reason its length is 2*N* rather than *N* is that the h_k's can be complex numbers. The real part of each number is stored first, then the complex part as shown in Fig. 10.45 (a). If the h_k's are real then the imaginary elements are simply set to zero.

The only output quantity from *FFT* is the array *Data*, which now contains the H_n's. These are also usually complex so again the real part of each H_n is stored first, then the complex part. Note carefully which frequency is stored at which location. The order is shown in Fig. 10.45(b). It is exactly the same order as in Fig. 10.43, namely DC first, then positive frequencies from lowest to highest, then negative frequencies from most negative to least negative.

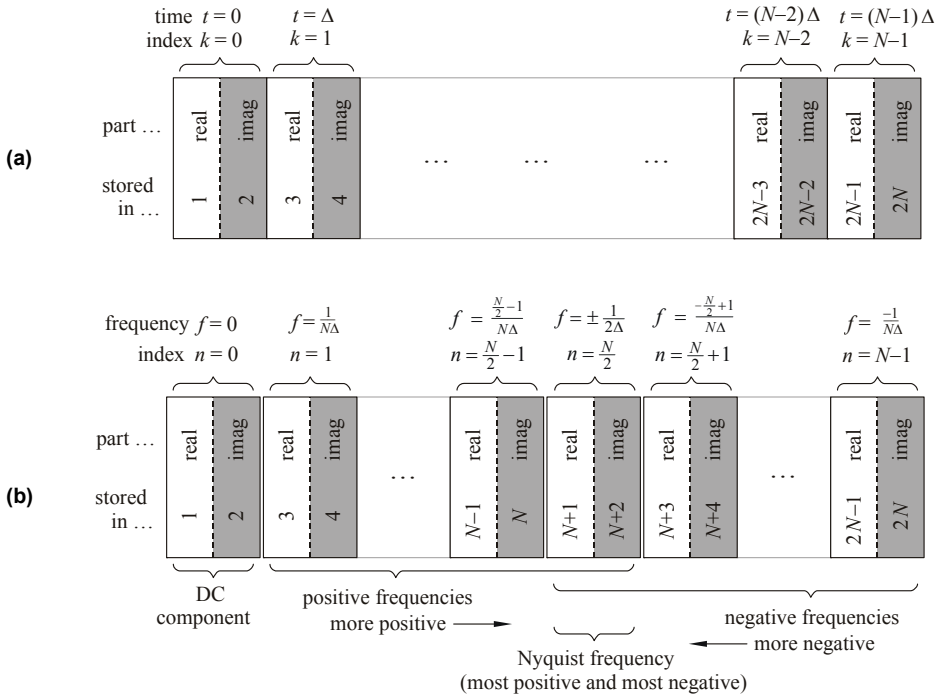


Figure 10.45 **(a)** Upon entry to routine *FFT*, the array *Data* holds the h_k's. The h_k's are generally complex numbers. The real part is stored first then the imaginary part. **(b)** Upon exit from routine *FFT*, array *Data* holds the H_n's. They are also complex. Their order is: DC, then positive frequencies, then negative; the same as in Fig. 10.43.

```
Sub FFT(Data() As Double, N As Integer, ISign As Integer)
   Const Pi = 3.14159265358979
   Dim I As Integer, IStep As Integer, J As Integer, M As Integer, MMax As Integer
   Dim NN As Integer, TempI As Double, TempR As Double,  WTemp As Double
   Dim Theta As Double, WI As Double, WPI As Double, WR As Double, WPR As Double

   NN = 2 * N
   J = 1

   For I = 1 To NN Step 2              'This is the bit-reversal section of the routine.
      If J > I Then
         TempR = Data(J)               'Exchange the two complex numbers.
         TempI = Data(J + 1)
         Data(J) = Data(I)
         Data(J + 1) = Data(I + 1)
         Data(I) = TempR
         Data(I + 1) = TempI
      End If

      M = N
1:    If M >= 2 And J > M Then
         J = J - M
         M = M / 2
         GoTo 1
      End If

      J = J + M
   Next I

   MMax = 2                            'This is the FFT splitting-in-two section of the routine.
2:  If NN > MMax Then                  'This outer loop is executed log2(N) times.
      IStep = 2 * MMax
      Theta = 2 * Pi / (ISign * MMax)  'Initialize for the trigonometric recurrence.
      WPR = -2 * Sin(0.5 * Theta) ^ 2
      WPI = Sin(Theta)
      WR = 1                           'WR and WI are real and imaginary parts of W
      WI = 0                           'raised to a power.

      For M = 1 To MMax Step 2         'Here are the two nested inner loops.
         For I = M To NN Step IStep
            J = I + MMax
            TempR = WR * Data(J) - WI * Data(J + 1)
            TempI = WR * Data(J + 1) + WI * Data(J)
            Data(J) = Data(I) - TempR
            Data(J + 1) = Data(I + 1) - TempI
            Data(I) = Data(I) + TempR
            Data(I + 1) = Data(I + 1) + TempI
         Next I

         WTemp = WR                    'Trigonometric recurrence for W raised to a new power.
         WR = WR * WPR - WI * WPI + WR
         WI = WI * WPR + WTemp * WPI + WI
      Next M

      MMax = IStep
      GoTo 2
   End If
End Sub
```

Example 10.15: (a) Write an Excel macro that uses the FFT subroutine listed on the previous page to approximate the Fourier transform of the function $h(t) = e^{-3t^2}\cos(6\pi t)$. Note that in Example 10.14 we found its exact Fourier transform. This example will illustrate the similarities and differences between the discrete Fourier transform (calculated here) and the continuous Fourier transform (calculated there).

(b) Suppose that $h(t)$ is the current flowing in a 1Ω resistor. Use the FFT to approximate the total energy dissipated in the resistor and the energy at each frequency (i.e. the power spectrum).

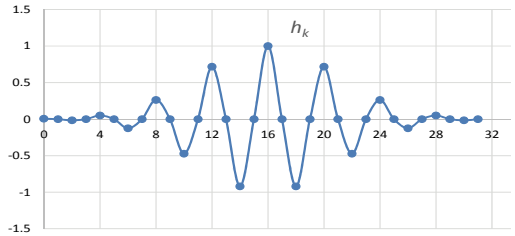
Solution: Here is a subroutine, *Wrapper*, written in Visual Basic, that sets up the h_k's, then calls *FFT*, and then prints the H_n's to an Excel spreadsheet. There is also a function, *Waveform*, which is the function to be Fourier transformed.

```
Public Sub Wrapper()
   Const ISign = 1
   Const N = 32                                        '[1] The number of sample points.
   Const DeltaT = 2.666667 / N
   Const Midpt = N / 2                                 'To achieve even or odd symmetry it is
   Dim Data() As Double                                'useful to put t=0 at Midpt index point.
   ReDim Data(1 To 2 * N)
   Dim K As Integer, Time As Double, Energy As Double, EngyT As Double, EngyF As Double

   For K = 0 To N - 1
      Time = -(K - Midpt) * DeltaT                     '[2] Note t=0 when k=Midpt
      Data(2 * K + 1) = Waveform(Time)                 '[3] Set up Re{hk}, the waveform.
      Data(2 * K + 2) = 0                              'Set up Im{hk}, all zeros.
      Cells(K + 1, 1) = K                              'Print the index k in col 1 of worksheet.
      Cells(K + 1, 2) = Data(2 * K + 1)                'Print hk in col 2.
      EngyT = EngyT + Data(2 * K + 1) ^ 2  * DeltaT    'Sum up Energy over all times.
   Next K

   Call FFT(Data, N, ISign)                            'DO THE FFT.

   For K = 0 To N - 1
      Cells(K + 1, 3) = Data(2 * K + 1)                'Print Re{Hn} in col 3.
      Cells(K + 1, 4) = Data(2 * K + 2)                'Print Im{Hn} in col 4.
   Next K

   EngyF = (Data(1) ^ 2 + Data(2) ^ 2) * DeltaT / N    'Calculate the energy in the DC term.
   Cells(1, 5) = EngyF                                 'Print DC energy in col 5.

   Energy = (Data(N + 1) ^ 2 + Data(N + 2) ^ 2) * DeltaT / N   'Calculate energy at Nyquist freq.
   Cells(N / 2 + 1, 5) = Energy                        'Print energy at Nyquist freq.
   EngyF = EngyF + Energy

   For K = 1 To N / 2 - 1                              'Calculate energy at all other frequencies.
      Energy = (Data(2 * K + 1) ^ 2 + Data(2 * K + 2) ^ 2 _
         + Data(2 * (N - K) + 1) ^ 2 + Data(2 * (N - K) + 2) ^ 2) *DeltaT / N
      Cells(K + 1, 5) = Energy                         'Print energy at all other frequencies.
      EngyF = EngyF + Energy                           'Sum up energy over all frequencies.
   Next K
End Sub

Function Waveform(T)                                   'The waveform to be transformed.
   Const Pi = 3.14159265358979
   Waveform = Exp(-3 * T ^ 2) * Cos(6 * Pi * T)
End Function
```

Copy *FFT*, *Wrapper* and *Waveform* into an Excel module (see the instructions on page 118 on how to do this). Because *Wrapper* is declared "public" it appears in the list of macros that can be run in Excel. Run it. It prints 5 columns of numbers into a spreadsheet:

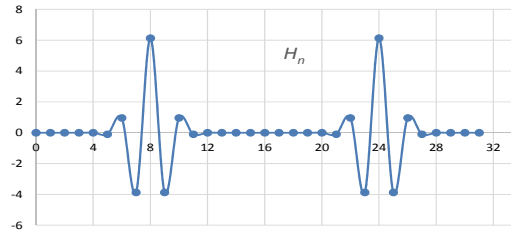
- Column 1 lists the index number k (or n). The number of data points is presently set to $N = 32$ (in the line labelled [1] in *Wrapper*), so this column contains the numbers 0 to 31.
- Column 2 lists the real parts of the h_k's. (The imaginary parts are zero and so not listed.) These values are calculated in line [3]. Line [2] is the connection between time t and the index k (compare with Eq. (10.85)). It causes time $t = 0$ to occur at index point $k = 16$ and the time interval between samples to be $\Delta = \frac{1}{12}$. Note that if the h_k's have even symmetry about the mid-index $k = 16$ (which is the case presently) then the H_n's will be real (i.e. the imaginary part of the Fourier transform will be zero). A scatter plot of column 2 vs. column 1 is shown in Fig. 10.46. (Note that the points are the h_k's and the curve is just a visual aid to help see the oscillations.) Compare it with Fig. 10.36.

Figure 10.46 The points are a plot of h_k, k=0, …, 31. The curve is a visual aid to help see the oscillations.



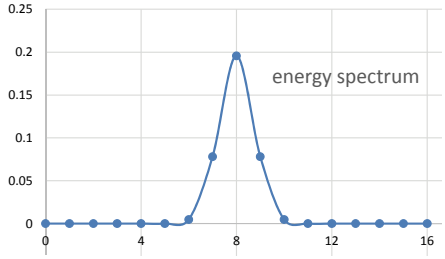
- Column 3 lists the real parts of the H_n's (the output from *FFT*). A scatter plot of column 3 vs. column 1 is shown in Fig. 10.47. It is interesting to compare it to Fig. 10.38 which is the exact Fourier transform.

Figure 10.47 The real part of H_n, n=0, …, 31. Again the curve is just a visual aid.



- Column 4 lists the imaginary parts of the H_n's. They are presently all zero. However if line [2] in *Wrapper* is changed to "Data(2 * K + 1) = Waveform(Time - 1/12)" then the waveform is shifted one index point to the right, the even symmetry is lost, and the imaginary parts of the H_n's become non-zero. This is very similar to what we saw on page 257, namely shifting a triangle waveform horizontally changed its Fourier series from cosines into sines. Here the real parts change into imaginary parts.
- Column 5 lists the energy spectrum, which is similar to the power spectrum for periodic waveforms. It is explained in more detail below. A plot of column 5 vs. column 1 is shown in Fig. 10.48.

Figure 10.48 The energy spectrum for n=0, …, 16.



Some background on the energy spectrum. Suppose that $h(t) = e^{-3t^2}\cos(6\pi t)$ is a current in amperes flowing through a 1Ω resistor. The instantaneous power is $p(t) = (h(t))^2 R$ and integrating this over all time gives the *total* energy, *E*, in joules dissipated in the resistor.

$$E = \int_{-\infty}^{\infty} (h(t))^2 \, dt = \int_{-\infty}^{\infty} \left(e^{-3t^2}\cos(6\pi t)\right)^2 dt = 0.3618 \, . \tag{10.95}$$

(This answer was gotten using Simpson's rule) Notice that it doesn't make sense to talk about the average power as we did in Chapter 6, Section 7, because here the period is infinite (the function never repeats) and the average power is zero. The discrete analog of (10.95) is

$$E = \int_{-\infty}^{\infty} (h(t))^2 \, dt \approx \sum_{k=0}^{N-1} (h_k)^2 \, \Delta \; = \; \Delta \cdot \sum_{k=0}^{N-1} |h_k|^2 \tag{10.96}$$

We can check the accuracy of this approximation by using the SUMSQ function in Excel to sum the squares of the numbers in column 2, giving 4.3416, and then multiplying by $\Delta = \frac{1}{12}$. To 4 sig. figs. this gives the same value as the Simpson integration did, namely 0.3618.

Using Parseval's theorem, (10.91), the total energy can also be expressed as a summation over frequencies.

$$E \approx \; \Delta \cdot \frac{1}{N} \sum_{n=0}^{N-1} |H_n|^2 \tag{10.97}$$

There are two important points when applying (10.97) to get the energy spectrum (the distribution of energy among the frequencies):

- Unlike the h_k's, the H_n's are generally complex numbers and for any complex number, $a + bj$, absolute value means $|a+bj|^2 = a^2 + b^2$. Thus, for example, to get E_0, the energy in the DC component, we have to use these elements of the array *Data* (see Fig. 10.45(b)):

  $$E_0 \approx \frac{\Delta}{N}|H_0|^2 = \frac{\Delta}{N}\left(Data(1)^2 + Data(2)^2\right)$$

  and to get $E_{N/2}$, the energy at the Nyquist frequency, we have to use these elements:

  $$E_{N/2} \approx \frac{\Delta}{N}|H_{N/2}|^2 = \frac{\Delta}{N}\left(Data(N+1)^2 + Data(N+2)^2\right)$$

- All frequencies other than the DC and the Nyquist frequency have both positive and negative frequency components that should be counted together. For example the energy in the lowest frequency, $f = \frac{1}{N\Delta}$, is

  $$E_1 \approx \frac{\Delta}{N}\left(|H_1|^2 + |H_{N-1}|^2\right) = \frac{\Delta}{N}\left(Data(3)^2 + Data(4)^2 + Data(2N-1)^2 + Data(2N)^2\right)$$

The other values in column 5 are gotten the same way. Notice that the energy spectrum is peaked at $n = 8$. The corresponding frequency, according to Eq. (10.86), is $f = \frac{n}{N\Delta} = \frac{8}{32 \cdot \frac{1}{12}} = 3$.